TDK InvenSense

ICP-10125 Datasheet

High Accuracy, Low Power, 10atm Waterproof Barometric Pressure and Temperature Sensor IC

GENERAL INFORMATION

The ICP-10125 pressure sensor is based on MEMS capacitive technology, which provides ultra-low noise at the lowest power, enabling industry leading relative accuracy, sensor throughput, and temperature stability. The pressure sensor can measure pressure differences with an accuracy of ±1 Pa, an accuracy enabling altitude measurement differentials as small as 8.5 cm, less than the height of a single stair step.

Consuming only 1.3 µA @1 Hz, the device is available in a small footprint 3.55 mm x 3.55 mm x 1.45 mm chimney package with waterproofing gel providing IPx8 waterproofing to 10 ATM. The ICP-10125 is ideally suited for wearable fitness monitoring and battery powered IoT.

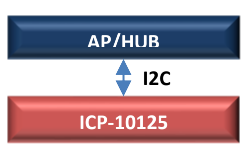
The ICP-10125 offers an industry leading temperature coefficient offset of ±0.5 Pa/°C. The combination of high accuracy, low power, temperature stability, waterproofing in a small footprint enables higher performance barometric pressure sensing for sports activity identification and mobile indoor/outdoor navigation.

DEVICE INFORMATION

PART NUMBER	PACKAGE	LID OPENING
ICP-10125	3.55x3.55x1.45mm HTCC-10L	Gel filled HTCC package with machined lid; IPx8 waterproofing to 10 ATM

Denotes RoHS and Green-Compliant Package

BLOCK DIAGRAMS



APPLICATIONS

- Smart watches
- Leisure, Sports, and Fitness Activity Monitoring for Wearable Sensors
- Altimeters and barometers for portable devices
- Indoor/Outdoor Navigation (dead-reckoning, floor/elevator/step detection)
- Home and Building Automation
- Weather Forecasting

FEATURES

- Pressure operating range: 30 to 110 kPa
- Noise and current consumption
  - 0.4 Pa @ 10.4 µA (ULN mode)
  - 0.8 Pa @ 5.2 µA (LN mode)
  - 3.2 Pa @ 1.3 µA (LP mode)
- Pressure Sensor Relative Accuracy: ±1 Pa for any 10 hPa change over 950 hPa-1050 hPa at 25°C
- Pressure Sensor Absolute Accuracy: ±1 hPa over 950 hPa-1050 hPa, 0°C to 65°C
- Pressure Sensor Temperature Coefficient Offset: ±0.5 Pa/°C over 25°C to 45°C at 100 kPa
- Temperature Sensor Absolute Accuracy: ±0.4°C
- IPx8: Waterproof to 10 ATM
- Temperature operating range: -40 °C to 85 °C
- Host Interface: I^2C at up to 400 kHz
- Single Supply voltage: 1.8V ±5%
- RoHS and Green compliant

TYPICAL OPERATING CIRCUIT

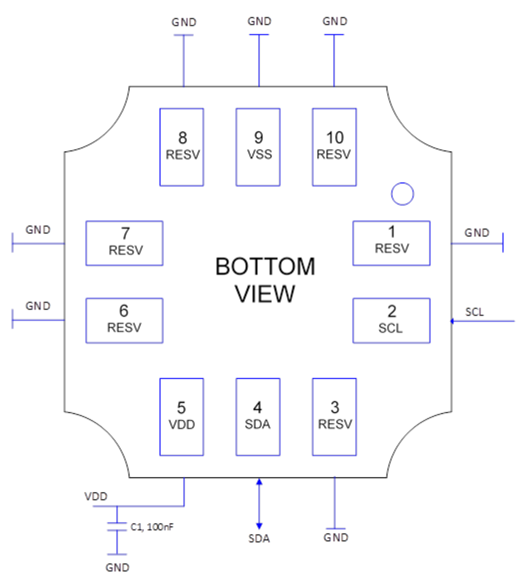


TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 PURPOSE AND SCOPE

This document is a preliminary product specification, providing a description, specifications, and design related information for the ICP-10125 Pressure Sensor.

Specifications are subject to change without notice. Final specifications will be updated based upon characterization of production silicon.

1.2 PRODUCT OVERVIEW

The ICP-10125 is an ultra-low power, low noise, digital output barometric pressure and temperature sensor IC. It is based on an innovative MEMS capacitive pressure sensor technology that can measure pressure differences with an accuracy of ±1 Pa at the industry's lowest power. The high accuracy MEMS capacitive pressure sensor is capable of measuring altitude differentials down to 8.5 cm without the penalty of increased power consumption or reduced sensor throughput.

The capacitive pressure sensor has a ±1 hPa absolute accuracy over its full range of 300 hPa -1100 hPa. The pressure sensor has an embedded temperature sensor and 400 kHz I^2C bus for communication. For power-critical applications, the ICP-10125 features a low power mode of 1.3 µA at a noise of 3.2 Pa or for high performance applications, it features a low noise mode of 0.8 Pa while only consuming 5.2 µA.

The device is available in a small footprint 3.55 mm x 3.55 mm x 1.45 mm chimney package with waterproofing gel providing IPx8 waterproofing to 10 ATM.

The ICP-10125 also offers industry leading temperature stability of the pressure sensor with a temperature coefficient offset of ±0.5 Pa/°C. The high accuracy, temperature stability, and market leading low power consumption of 1.3 µA @1 Hz offered by ICP-10125 makes it ideally suited for applications such as mobile phones, drone flight control and stabilization, indoor/outdoor navigation (elevator, floor, and stair step detection), sports and fitness activity monitoring, and battery-powered IoT.

2 PRESSURE AND TEMPERATURE SENSOR SPECIFICATIONS

2.1 OPERATION RANGES

The sensor shows best performance when operated within the recommended temperature and pressure range (hereafter called normal conditions) of 0°C – 45°C and 95 kPa – 105 kPa, respectively. The following ranges are defined for the device:

OPERATION RANGE	PRESSURE (KPA)	TEMPERATURE (°C)
Normal	95 to 105	0 to 45
Extended	30 to 110	-20 to 65
Maximum	25 to 115	-40 to 85

Table 1. Operation Ranges

2.2 OPERATION MODES

The sensor can be operated in up to four different measurement modes to satisfy different requirements for power consumption vs. noise, accuracy, and measurement frequency. An overview of the operation modes is given in Table 2.

PARAMETER	CONDITIONS	SENSOR MODE	TYP	MAX	UNITS	NOTES
Conversion Time	Time between sending last bit of measurement command, and sensor data ready for measurement	Low Power (LP)	1.6	1.8	ms	1
		Normal (N)	5.6	6.3		1
		Low Noise (LN)	20.8	23.8		1
		Ultra Low Noise (ULN)	83.2	94.5		1
Current Consumption	1 Hz ODR	Low Power (LP)	1.3		µA	
		Normal (N)	2.6			
		Low Noise (LN)	5.2			
		Ultra Low Noise (ULN)	10.4			
Pressure RMS Noise	Valid for P = 100 kPa, T = 25°C, and U = 1.8V	Low Power (LP)	3.2		Pa	
		Normal	1.6			
		Low Noise (LN)	0.8			
		Ultra Low Noise (ULN)	0.4			

Table 2. Operation Modes

Notes:

1. Guaranteed by design.

Low Power modes supports ODR greater than 500 Hz while the Low Noise mode provides industry leading RMS noise at a fast 40 Hz ODR. Further decrease in noise may be achieved by software oversampling and filtering through customer’s software implementation or custom TDK InvenSense operation modes available upon request.

2.3 PRESSURE SENSOR SPECIFICATIONS

Pressure sensor specifications are given in Table 3. Default conditions of 25 °C and 1.8V supply voltage apply, unless otherwise stated.

PARAMETER	CONDITIONS	TYP	UNITS	NOTES
Absolute Accuracy	Normal range Extended range	±1 ±1.5	hPa	1
Relative Accuracy	Any step ≤ 1 kPa, 25 °C Any step ≤ 10 kPa, 25 °C	±1 ±3	Pa	
Long-term drift **During 1 year**	Normal range Extended range	±35 ±40	Pa/y	
Solder drift		1.5	hPa	1, 2
Temperature coefficient offset	P = 100 kPa 25°C … 45°C	±0.5	Pa/°C	
Resolution	Maximum range	0.01	Pa	

Table 3. Pressure Sensor Specifications

Notes:

1. Absolute accuracy may be improved through One Point Calibration
2. Sensor accuracy post Solder reflow may be improved through One Point Calibration

2.4 TEMPERATURE SENSOR SPECIFICATIONS

Specifications of the temperature sensor are shown in Table 4.

PARAMETER	CONDITIONS	TYP	UNITS
Absolute Accuracy	Extended range	±0.4	°C
Repeatability	Extended range	±0.1	°C
Resolution	Maximum range	0.01	°C
Long-term drift	Normal range	<0.04	°C/y

Table 4. Temperature Sensor Specifications

2.5 RECOMMENDED OPERATION CONDITIONS

The pressure sensor exhibits best performance when operated within the normal pressure and temperature range 0°C < T < 45°C and 95 kPa < P < 105 kPa.

Injected photo current due to strong light sources can influence the sensor performance and should be avoided to guarantee best operation.

The sensor should not be exposed to high mechanical stress, the resulting deformation of the package can alter internal dimensions and therefore falsify the sensor signal. Solder reflow may affect device performance. One-point calibration can improve the sensor accuracy post solder reflow.

3 ELECTRICAL SPECIFICATIONS

3.1 ELECTRICAL CHARACTERISTICS

Default conditions of 25 °C and 1.8V supply voltage apply to values in Table 5, unless otherwise stated.

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS	COMMENTS
Supply voltage	V_{DD}		1.71	1.8	1.89	V	
Power-up/down level	V_{POR}	Static power supply	1.0	1.25	1.5	V	
Supply Ramp Time	T_{RAMP}	Monotonic ramp. Ramp rate is 10% to 90% of the final value	0.01		100	ms	
Supply current	I_{DD}	Idle state	-	1.0	2.5	µA	
		Measurement	-	210	300	µA	Current consumption while sensor is measuring.
		Average	-	1.3	-	µA	Current consumption in continuous operation @ 1 Hz ODR in LP Mode
			-	5.2	-	µA	Current consumption in continuous operation @1 Hz ODR in LN Mode
Low level input voltage	V_{IL}		0	-	0.3 V_{DD}	V	
High level input voltage	V_{IH}		0.7 V_{DD}	-	V_{DD}	V	
Low level output voltage	V_{OL}	0 < IOL < 3 mA	-	-	0.2 V_{DD}	V	
Output Sink Current	I_{OL}	V_{OL} = 0.4V	3.1	4.1	-	mA	
		V_{OL} = 0.6V	3.5	4.5	-	mA	

Table 5. Electrical Specifications

3.2 ABSOLUTE MAXIMUM RATINGS

Stress levels beyond those listed in Table 6 may cause permanent damage to the device. These are stress ratings only and functional operation of the device at these conditions cannot be guaranteed. Exposure to the absolute maximum rating conditions for extended periods may affect the reliability of the device.

PARAMETER	RATING
Supply voltage, VDD	-0.3V to 2.16V
Supply Voltage, SCL & SDA	-0.3V to VDD 0.3V
Operating temperature range	-40°C to 85°C
Storage temperature range	-40°C to 125°C
ESD HBM	2.0 kV
ESD CDM	250V
Latch up, JESD78 Class II, 85°C	100 mA
Overpressure	>600kPa

Table 6. Absolute Maximum Ratings

3.3 SENSOR SYSTEM TIMING

Default conditions of 25°C and 1.8V supply voltage apply to typ. values listed in Table 7, unless otherwise stated. Max. values apply over the specified operating range of VDD and over the operating temperature range.

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS	COMMENTS
Power-up time	t_{PU}	After hard reset, $V_{DD} \geq V_{POR}$	-	170	-	µs	Time between V_{DD} reaching V_{PU} and sensor entering idle state
Soft reset time	t_{SR}	After soft reset	-	170	-	µs	Time between ACK of soft reset command and sensor entering idle state
Measurement duration	t_{MEAS}	LN Mode	-	20.8	23.8	ms	Duration for a pressure and temperature measurement

Table 7. System Timing Specifications

3.4 I²C TIMING CHARACTERIZATION

Default conditions of 25°C and 1.8V supply voltage apply to values in Table 8, unless otherwise stated.

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
SCL clock frequency	f_{SCL}		0	-	400	kHz
Hold time (repeated) START condition	$t_{HD;STA}$	After this period, the first clock pulse is generated	0.6	-	-	µs
LOW period of the SCL clock	t_{LOW}		1.3	-	-	µs
HIGH period of the SCL clock	t_{HIGH}		0.6	-	-	µs
Set-up time for a repeated START condition	$t_{SU;STA}$		0.6	-	-	µs
SDA hold time	$t_{HD;DAT}$		0	-	-	µs
SDA set-up time	$t_{SU;DAT}$		100	-	-	ns
SCL/SDA rise time	t_R		20	-	300	ns
SCL/SDA fall time	t_F		-	-	300	ns
SDA valid time	$t_{VD;DAT}$		-	-	0.9	µs
Set-up time for STOP condition	$t_{SU;STO}$		0.6	-	-	µs
Capacitive load on bus line	C_B		-	-	400	pF

Table 8. I²C Parameters Specification

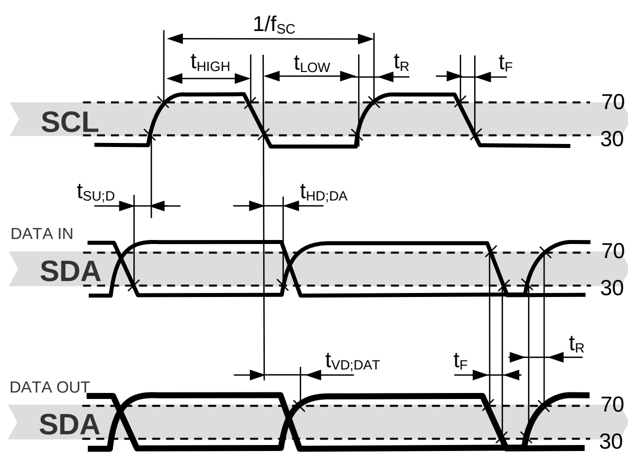


Figure 1. Digital I/O Pads Timing

4 APPLICATIONS INFORMATION

4.1 INTERFACE SPECIFICATIONS

The ICP-10125 supports I²C fast mode, SCL clock frequency from 0 to 400 kHz.

4.2 PIN OUT DIAGRAM AND SIGNAL DESCRIPTION

PIN NUMBER	PIN NAME	DESCRIPTION
1	RESV	No Connect (NC) or Connect to GND
2	SCL	I²C Serial Clock
3	RESV	Connect to Ground
4	SDA	I²C Serial Data
5	VDD	Power Supply VDD
6	RESV	No Connect (NC) or Connect to GND
7	RESV	No Connect (NC) or Connect to GND
8	RESV	Connect to Ground
9	VSS	Connect to Ground
10	RESV	No Connect (NC) or Connect to GND

Table 9. Signal Descriptions

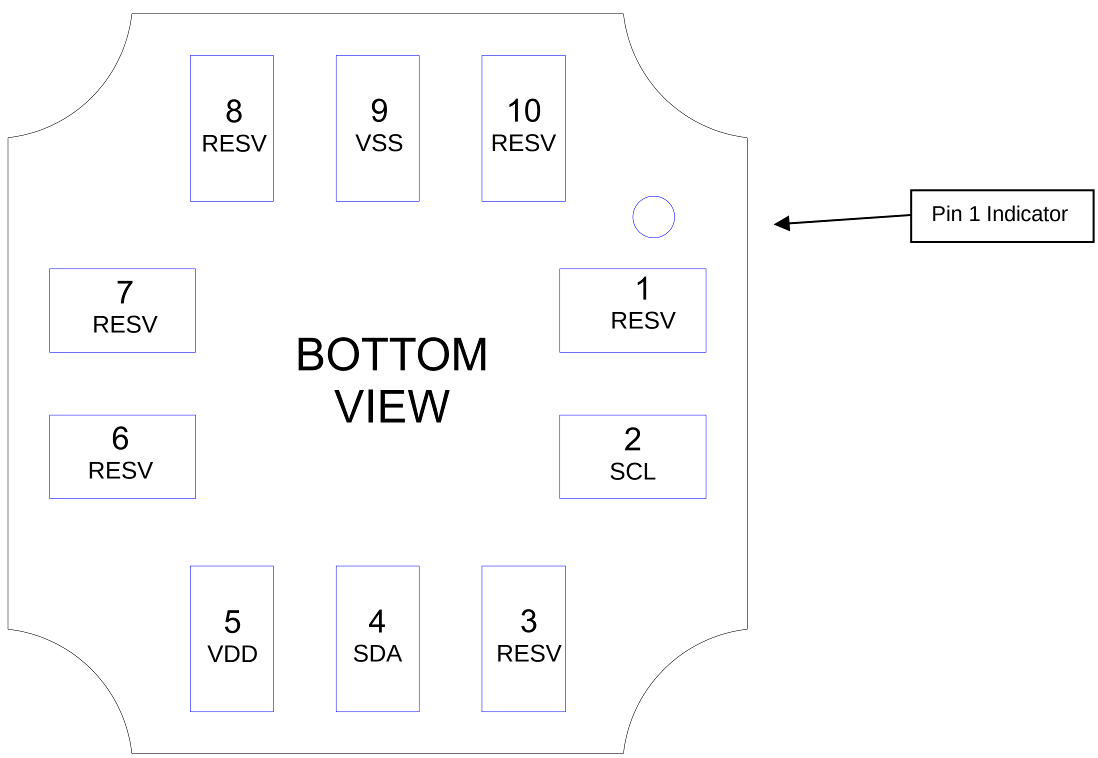


Figure 2. Pin Out Diagram for ICP-10125, 3.55mm x 3.55mm x 1.45mm HTCC

4.3 TYPICAL OPERATING CIRCUIT

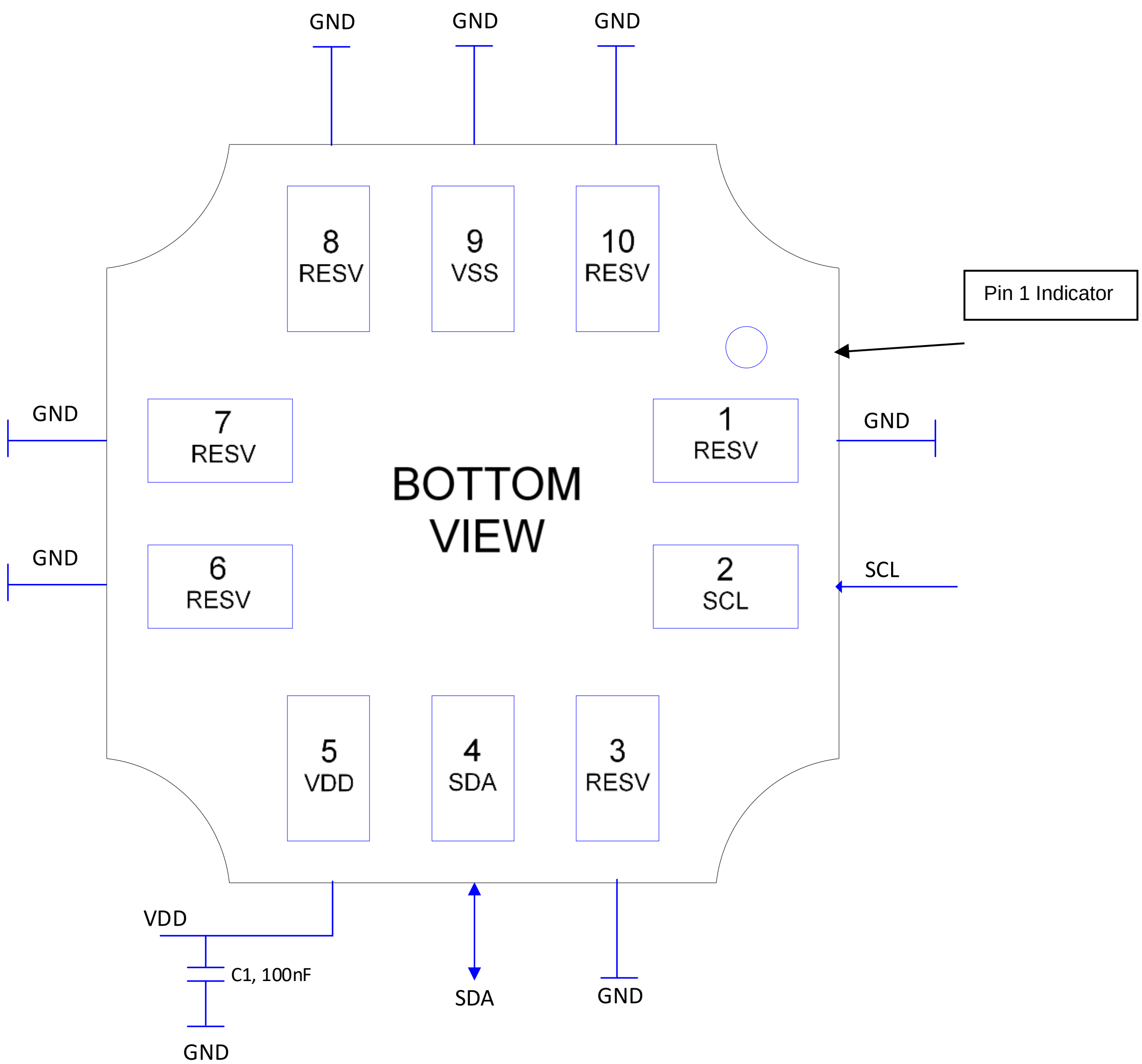
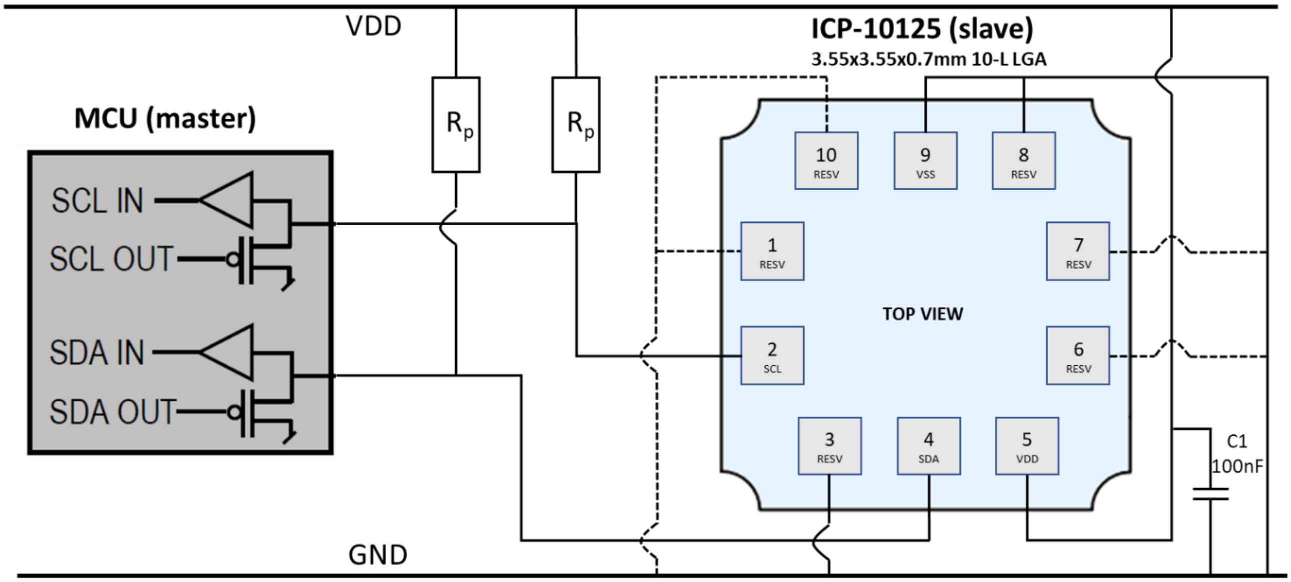


Figure 3. ICP-10125 Application Schematic

Power supply pins supply voltage (VDD) and ground (VSS) must be decoupled with a 100 nF capacitor that shall be placed as close to the sensor as possible. Connections shown as dashed lines are recommended for mechanical stability of the sensor (see Figure 4).



Connections shown as dashed lines are recommended for mechanical stability of the sensor

Figure 4. Typical Application Circuit

SCL is used to synchronize the communication between the microcontroller and the sensor. The master must keep the clock frequency within 0 to 400 kHz as specified in Table 8.

The SDA pin is used to transfer data in and out of the sensor. For safe communication, the timing specifications defined in the I^2C manual must be met.

To avoid signal contention, the microcontroller must only drive SDA and SCL low. External pull-up resistors (i.e. 10 kΩ) are required to pull the signal high. For dimensioning resistor sizes, user should also consider bus capacity requirements. It should be noted that pull-up resistors may be included in I/O circuits of microcontrollers.

4.4 BILL OF MATERIALS FOR EXTERNAL COMPONENTS

COMPONENT	LABEL	SPECIFICATION	QUANTITY
VDD Bypass Capacitor	C1	Ceramic, X7R, 100 nF ±10%	1

Table 10. Bill of Materials

5 OPERATION AND COMMUNICATION

All commands and memory locations of the ICP-10125 are mapped to a 16-bit address space which can be accessed via the I²C protocol.

ICP-10125	BINARY	DECIMAL	HEXADECIMAL
I²C address	110'0011	99	0x63

Table 11. ICP-10125 I²C Device Address

5.1 POWER-UP AND COMMUNICATION START

When VDD reaches the power-up voltage level V_{POR}, the ICP-10125 enters idle state after a duration of t_{PU}. In idle state, the ICP-10125 is ready to receive commands from the master (microcontroller).

Each transmission sequence begins with START condition (S) and ends with an (optional) STOP condition (P) as described in the I²C-bus specification. Whenever the sensor is powered up, but not performing a measurement or communicating, it automatically enters idle state for energy saving.

5.2 MEASUREMENT COMMANDS

The ICP-10125 provides the possibility to define the sensor behavior during measurement as well as the transmission sequence of measurement results. These characteristics are defined by the appropriate measurement command.

Each measurement command triggers both a temperature and a pressure measurement.

OPERATION MODE	TRANSMIT T FIRST	TRANSMIT P FIRST
Low Power (LP)	0x609C	0x401A
Normal (N)	0x6825	0x48A3
Low Noise (LN)	0x70DF	0x5059
Ultra-Low Noise (ULN)	0x7866	0x58E0

Table 12. Measurement Commands

5.3 STARTING A MEASUREMENT

A measurement communication sequence consists of a START condition followed by the I²C header with the 7-bit I²C device address and a write bit (write W: '0', 8-bit word including I²C header: 0xC6). The sensor indicates the proper reception of a byte by pulling the SDA pin low (ACK bit) after the falling edge of the 8th SCL clock. Then the sensor is ready to receive a 16-bit measurement command. Again, the ICP-10125 acknowledges the proper reception of each byte with ACK condition. A complete measurement cycle is presented in Figure 5.

With the acknowledgement of the measurement command, the ICP-10125 starts measuring pressure and temperature.

5.4 SENSOR BEHAVIOR DURING MEASUREMENT

In general, the sensor does not respond to any I²C activity during measurement, i.e. I²C read and write headers are not acknowledged (NACK).

5.5 READOUT OF MEASUREMENT RESULTS

After a measurement command has been issued and the sensor has completed the measurement, the master can read the measurement results by sending a START condition followed by an I^2C read header (8-bit word including I^2C header: 0xC7). The sensor will acknowledge the reception of the read header and send the measured data in the specified order to the master. The MSB of the corresponding data is always transmitted first. Temperature data is transmitted in two 8-bit words and pressure data is transmitted in four 8-bit words. Regarding the pressure data, only the first three words MMSB, MLSB and LMSB contain information about the ADC pressure value *p_dout*. Therefore, for retrieving the ADC pressure value, LLSB must be disregarded:

$$p\_dout = \text{MMSB} \ll 16 \mid \text{MLSB} \ll 8 \mid \text{LMSB}.$$

Two bytes of data are always followed by one byte CRC checksum, for calculation see section 5.8. Each byte must be acknowledged by the microcontroller with an ACK condition for the sensor to continue sending data. If the ICP-10125 does not receive an ACK from the master after any byte of data, it will not continue sending data.

Whether the sensor sends out pressure or temperature data first depends on the measurement command that was sent to the sensor to initiate the measurement (see Table 12).

The I^2C master can abort the read transfer with a NACK condition after any data byte if it is not interested in subsequent data, e.g. the CRC byte or the second measurement result, to save time.

5.6 SOFT RESET

The ICP-10125 provides a soft reset mechanism that forces the system into a well-defined state without removing the power supply. If the system is in idle state (i.e. if no measurement is in progress) the soft reset command will be accepted by ICP-10125. This triggers the sensor to reset all internal state machines and reload calibration data from the memory.

COMMAND	HEXADECIMAL CODE	BINARY CODE
Soft reset	0x805D	1000’0000’0101’1101

Table 13. Soft Reset Command

5.7 READ-OUT OF ID REGISTER

The ICP-10125 has an ID register which contains a specific product code. The read-out of the ID register can be used to verify the presence of the sensor and proper communication. The command to read the ID register is shown in Table 14.

COMMAND	HEXADECIMAL CODE	BINARY CODE
Read ID register	0xEFC8	1110’1111’1100’1000

Table 14. Read-Out Command of ID Register

It needs to be sent to the ICP-10125 after an I^2C write header. After the ICP-10125 has acknowledged the proper reception of the command, the master can send an I^2C read header and the ICP-10125 will submit the 16-bit ID followed by 8 bits of CRC. The structure of the ID is described in Table 15. Bits 15:6 of the ID contain unspecified information (marked as “x”), which may vary from sensor to sensor, while bits 5:0 contain the ICP-10125 specific product code.

16-bit ID

xxxx'xxxx'xx 00'1000

bits 5 to 0: ICP-10125-specific product code

bits 15 to 6: unspecified information

Table 15. 16-bit ID Structure

5.8 CHECKSUM CALCULATION

The 8-bit CRC checksum transmitted after each data word is generated by a CRC algorithm with the properties displayed in Table 16. The CRC covers the contents of the two previously transmitted data bytes.

PROPERTY	VALUE
Name	CRC-8
Width	8 bits
Polynomial	0x31 ($x^8 + x^5 + x^4 + 1$)
Initialization	0xFF
Reflect input	false
Reflect output	false
Final XOR	0x00
Examples	CRC(0x00) = 0xAC CRC(0xBEEF) = 0x92

Table 16. ICP-10125 I²C CRC Properties

5.9 CONVERSION OF SIGNAL OUTPUT

Pressure measurement data is always transferred as 4 8-bit words; temperature measurement data is always transferred as two 8-bit words. Please see section 5.5 for more details.

Temperature measurement values *t_dout* are linearized by the ICP-10125 and must be calculated to °C by the user via the following formula:

$$T = -45°\mathrm{C} + \frac{175°\mathrm{C}}{2^{16}} \times t\_dout.$$

For retrieving physical pressure values in Pa the following conversion formula has to be used:

$$P = A + \frac{B}{C + p_{\mathrm{dout}}},$$

where *p_dout* is the sensor's raw pressure output. The converted output is compensated for temperature effects via the temperature dependent functions *A*, *B* and *C*. Besides the raw temperature output *t_dout*, the calculation of *A*, *B* and *C* requires to access calibration parameters OTP0, OTP1, OTP2, OTP3 stored in the OTP of the sensor. Read-out of OTP parameters is described in section 5.10.

Full sample code for calculating physical pressure values is given in section 5.11. The general workflow of the conversion is done by:

1) Import class Invensense_pressure_conversion
2) Read out values OTP0, …, OTP3 and save to *c1*, …, *c4*
3) Create object *name* for an individual sensor with parameter values *c1*, …, *c4*
   name = Invensense_pressure_conversion
   ([c1,c2,c3,c4])
4) Get raw pressure *p_dout* and temperature *t_dout* data from the sensor as described in chapter 5.5.
5) Call function get_pressure:
   name.get_pressure(p_dout, t_dout)

The sample code from section 5.13 gives an example of this workflow.

5.10 READ-OUT OF CALIBRATION PARAMETERS

For converting raw pressure data to physical values, four calibration parameters have to be retrieved from the OTP of the sensor.

Set up of OTP read:

1) Send I^2C write header 0xC6
2) Send command 0xC595 (move pointer in address register)
3) Send address parameter together with its CRC 0x00669C

Steps 1) – 3) can be executed on many platforms by a single I^2C write of the value 0xC59500669C.

Read out parameters:

Repeat the following procedure 4 times:

a) Send I^2C write header 0xC6
b) Send command 0xC7F7 (incremental read-out of OTP)
c) Send I^2C read header 0xC7
d) Read 3B (2B of data and 1B of CRC)
e) Decode data as 16bit big endian signed integer and store result into n-th calibration parameter *cn*.

Steps a) to d) can be executed on many platforms by a single write 0xC7F7 to the chip address followed by a single read of 3 B from the chip address.

5.11 SAMPLE CODE: EXAMPLE C SYNTAX

```
/* data structure to hold pressure sensor related parameters */
typedef struct inv_invpres
{
  struct inv_invpres_serif serif;
  uint32_t min_delay_us;
  uint8_t pressure_en;
  uint8_t temperature_en;
  float sensor_constants[4]; // OTP values
  float p_Pa_calib[3];
  float LUT_lower;
  float LUT_upper;
  float quadr_factor;
  float offst_factor;
} inv_invpres_t;


int inv_invpres_init(struct inv_invpres * s)
{
  short otp[4];

  read_otp_from_i2c(s, otp);

 init_base(s, otp);

  return 0;
}

int read_otp_from_i2c(struct inv_invpres * s, short *out)
{
  unsigned char data_write[10];
  unsigned char data_read[10] = {0};
  int status;
  int i;

  // OTP Read mode
```

```
	data_write[0] = 0xC5;
	data_write[1] = 0x95;
	data_write[2] = 0x00;
	data_write[3] = 0x66;
	data_write[4] = 0x9C;
	status = inv_invpres_serif_write_reg(&s->serif, ICC_ADDR_PRS, data_write, 5);
	if (status)
		return status;

	// Read OTP values
	for (i = 0; i < 4; i++) {
		data_write[0] = 0xC7;
		data_write[1] = 0xF7;
		status = inv_invpres_serif_write_reg(&s->serif, ICC_ADDR_PRS, data_write, 2);
		if (status)
			return status;

		status = inv_invpres_serif_read_reg(&s->serif, ICC_ADDR_PRS, data_read, 3);
		if (status)
			return status;

		out[i] = data_read[0]<<8 | data_read[1];
	}

	return 0;

}

void init_base(struct inv_invpres * s, short *otp)
{
	int i;

	for(i = 0; i < 4; i++)
		s->sensor_constants[i] = (float)otp[i];

	s->p_Pa_calib[0] = 45000.0;
	s->p_Pa_calib[1] = 80000.0;
	s->p_Pa_calib[2] = 105000.0;
	s->LUT_lower = 3.5 * (1<<20);
	s->LUT_upper = 11.5 * (1<<20);
	s->quadr_factor = 1 / 16777216.0;
	s->offst_factor = 2048.0;

}

// p_LSB -- Raw pressure data from sensor
// T_LSB -- Raw temperature data from sensor
int inv_invpres_process_data(struct inv_invpres * s, int p_LSB, int T_LSB,
							float * pressure, float * temperature)
{
	float t;
	float s1,s2,s3;
	float in[3];
	float out[3];
	float A,B,C;

	t = (float)(T_LSB - 32768);
	s1 = s->LUT_lower + (float)(s->sensor_constants[0] * t * t) * s->quadr_factor;
	s2 = s->offst_factor * s->sensor_constants[3] + (float)(s->sensor_constants[1] * t * t) * s->quadr_factor;
	s3 = s->LUT_upper + (float)(s->sensor_constants[2] * t * t) * s->quadr_factor;
	in[0] = s1;
	in[1] = s2;
	in[2] = s3;

	calculate_conversion_constants(s, s->p_Pa_calib, in, out);
	A = out[0];
	B = out[1];
	C = out[2];

	*pressure = A + B / (C + p_LSB);
	*temperature = -45.f + 175.f/65536.f * T_LSB;

	return 0;
}

// p_Pa -- List of 3 values corresponding to applied pressure in Pa
// p_LUT -- List of 3 values corresponding to the measured p_LUT values at the applied pressures.
void calculate_conversion_constants(struct inv_invpres * s, float *p_Pa,
								float *p_LUT, float *out)
```

```
{
  float A,B,C;

  C = (p_LUT[0] * p_LUT[1] * (p_Pa[0] - p_Pa[1]) +
          p_LUT[1] * p_LUT[2] * (p_Pa[1] - p_Pa[2]) +
          p_LUT[2] * p_LUT[0] * (p_Pa[2] - p_Pa[0])) /
          (p_LUT[2] * (p_Pa[0] - p_Pa[1]) +
          p_LUT[0] * (p_Pa[1] - p_Pa[2]) +
          p_LUT[1] * (p_Pa[2] - p_Pa[0]));
  A = (p_Pa[0] * p_LUT[0] - p_Pa[1] * p_LUT[1] - (p_Pa[1] - p_Pa[0]) * C) / (p_LUT[0] - p_LUT[1]);
  B = (p_Pa[0] - A) * (p_LUT[0] + C);

  out[0] = A;
  out[1] = B;
  out[2] = C;
}
```

5.12 SAMPLE CODE: CONVERSION FORMULA (EXAMPLE PYTHON SYNTAX)

```
class  InvensensePressureConversion:
    """ Class for conversion of the pressure and temperature output of the Invensense sensor"""


    def __init__(self, sensor_constants):
        """ Initialize customer formula


        Arguments:
        sensor_constants -- list of 4 integers: [c1, c2, c3, c4]
        """
        self.sensor_constants = sensor_constants


        # configuration for ICP-10125 Samples
        self.p_Pa_calib = [45000.0, 80000.0, 105000.0]
        self.LUT_lower = 3.5 * (2**20)
        self.LUT_upper = 11.5 * (2**20)
        self.quadr_factor = 1 / 16777216.0
        self.offst_factor = 2048.0


    def calculate_conversion_constants(self, p_Pa, p_LUT):
        """ calculate temperature dependent constants


        Arguments:
        p_Pa -- List of 3 values corresponding to applied pressure in Pa
        p_LUT -- List of 3 values corresponding to the measured p_LUT values at the applied pressures.
        """


        C = (p_LUT[0] * p_LUT[1] * (p_Pa[0] - p_Pa[1]) +
             p_LUT[1] * p_LUT[2] * (p_Pa[1] - p_Pa[2]) +
             p_LUT[2] * p_LUT[0] * (p_Pa[2] - p_Pa[0])) / \
            (p_LUT[2] * (p_Pa[0] - p_Pa[1]) +
             p_LUT[0] * (p_Pa[1] - p_Pa[2]) +
             p_LUT[1] * (p_Pa[2] - p_Pa[0]))
        A = (p_Pa[0] * p_LUT[0] - p_Pa[1] * p_LUT[1] - (p_Pa[1] - p_Pa[0]) * C) / (p_LUT[0] - p_LUT[1])
```

```
        B = (p_Pa[0] - A) * (p_LUT[0] + C)

        return [A, B, C]

    def get_pressure(self, p_LSB, T_LSB):
        """ Convert an output from a calibrated sensor to a pressure in Pa.

        Arguments:
        p_LSB -- Raw pressure data from sensor
        T_LSB -- Raw temperature data from sensor
        """
        t = T_LSB - 32768.0
        s1 = self.LUT_lower + float(self.sensor_constants[0] * t * t) * self.quadr_factor
        s2 = self.offst_factor * self.sensor_constants[3] + float(self.sensor_constants[1] * t * t) * self.quadr_factor
        s3 = self.LUT_upper + float(self.sensor_constants[2] * t * t) * self.quadr_factor
        A, B, C = self.calculate_conversion_constants(self.p_Pa_calib, [s1, s2, s3])
        return A + B / (C + p_LSB)
[end of the pseudocode]
```

5.13 SAMPLE CODE: USING CONVERSION FORMULA (EXAMPLE PYTHON SYNTAX)

```
def read_otp_from_i2c():
          # TODO: implement read from I2C
          # refer to data sheet for I2C commands to read OTP
          return 1000, 2000, 3000, 4000

def read_raw_pressure_temp_from_i2c():
          # TODO: implement read from I2C
          # refer to data sheet for I2C commands to read pressure and temperature
          return 8000000, 32000

# Sample code to read
from  Invensense_pressure_conversion import  Invensense_pressure_conversion

# -- initialization
c1, c2, c3, c4 = read_otp_from_i2c()
conversion =  Invensense_pressure_conversion([c1, c2, c3, c4])

# -- read raw pressure and temp data, calculate pressure
p, T = read_raw_pressure_temp_from_i2c()
pressure = conversion.get_pressure(p, T)
print 'Pressure: %f' % pressure
[end of the pseudocode]
```

5.14 COMMUNICATION DATA SEQUENCES

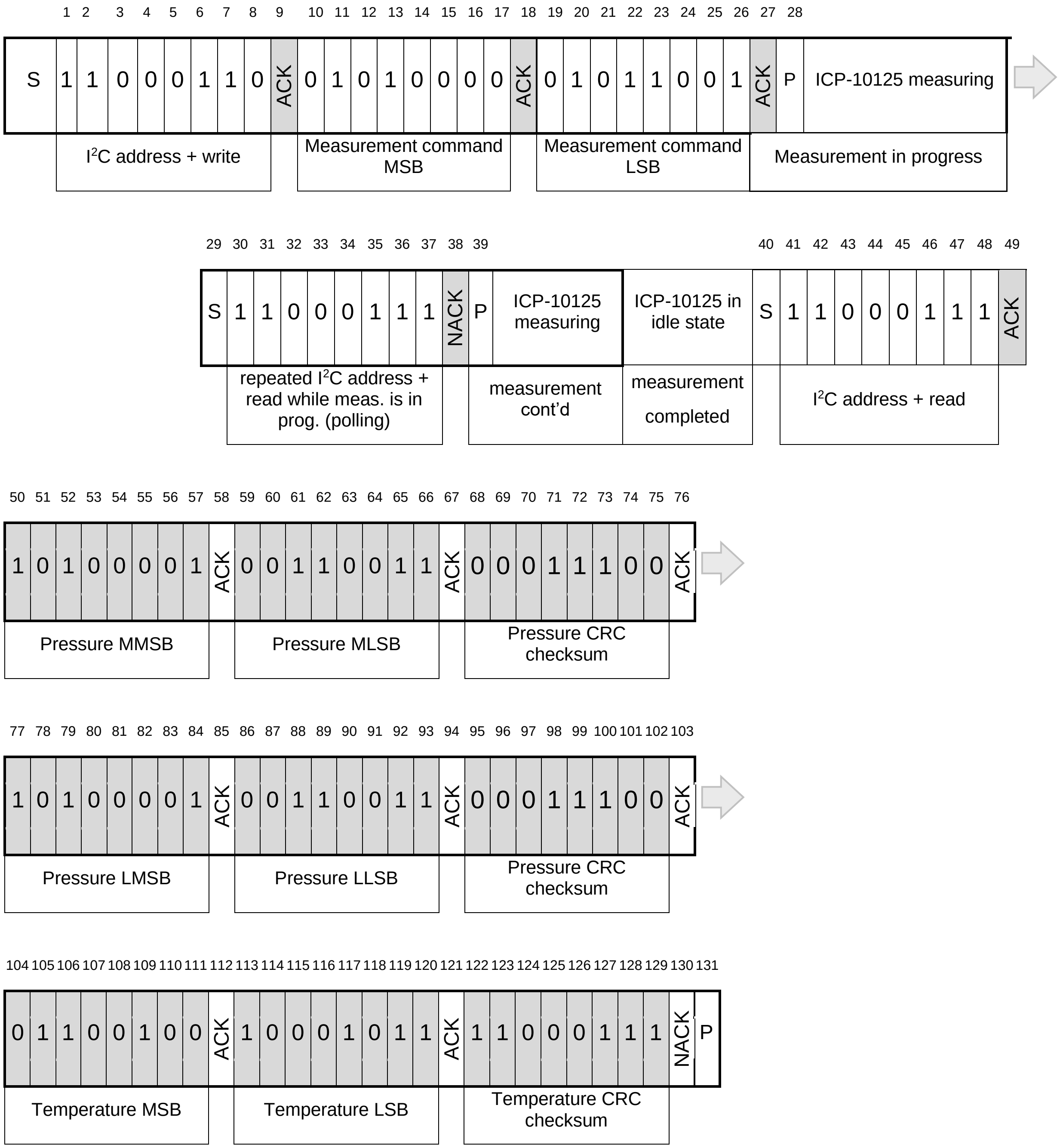


Figure 5. Communication Data Sequences

6 ASSEMBLY

This section provides general guidelines for assembling TDK-InvenSense Micro Electro-Mechanical Systems (MEMS) pressure sensors.

6.1 IMPLEMENTATION AND USAGE RECOMMENDATIONS

6.1.1 Soldering

When soldering, use the standard soldering profile IPC/JEDEC J-STD-020 with peak temperatures of 260°C. ICP-10125 may exhibit a pressure offset after soldering, some settling time may be required depending on soldering properties, PCB properties, and ambient conditions.

The ICP-10125 package consists of a chimney port that opens to the sensing element. Special care must be taken during soldering process to avoid contaminating the sensor through the open chimney.

1. Solder the sensor as a second soldering operation, after other components have been soldered
2. Use No-Clean solder paste
3. Sensor must not be subjected to board washing of any kind (critical)

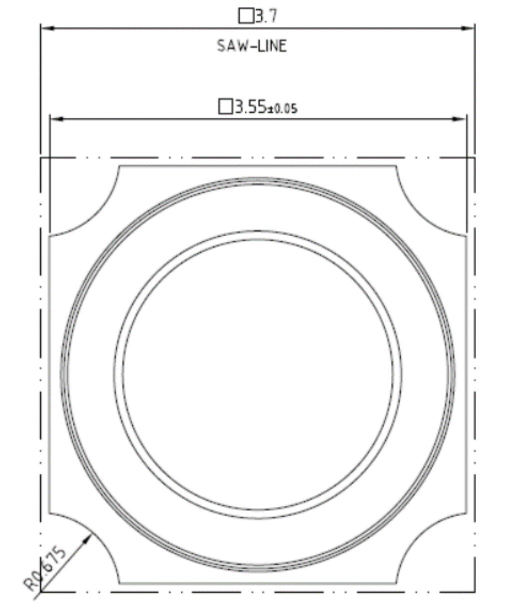
6.1.2 Chemical Exposure and Sensor Protection

The ICP-10125 must not be exposed to particulates or liquids. If any type of protective coating must be applied to the circuit board, the sensor must be protected during the coating process.

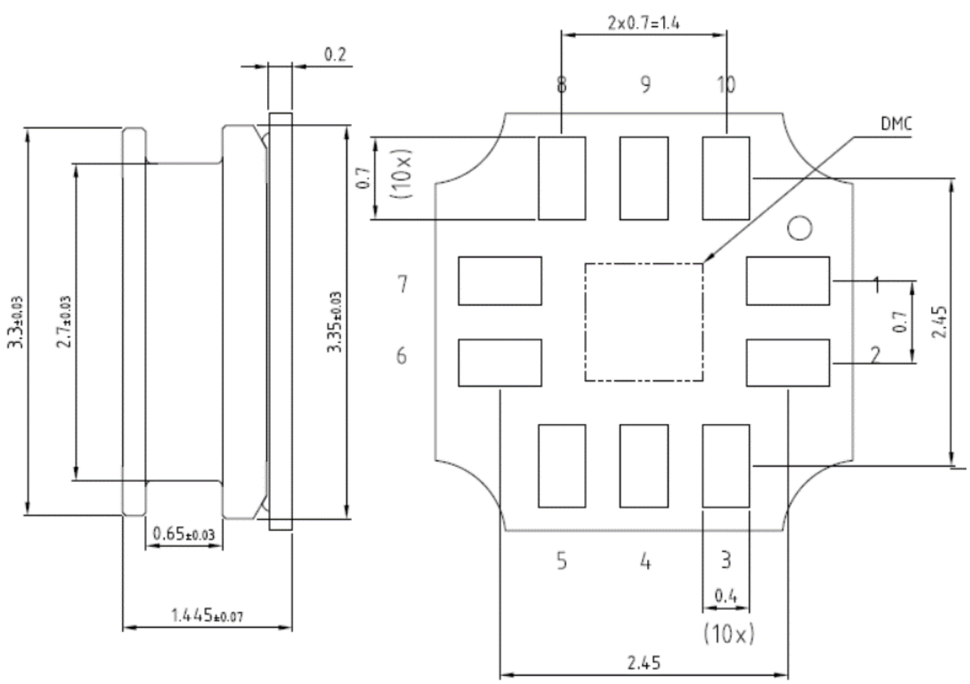
For further information on assembly, please refer to AN-000140 *TDK-InvenSense Pressure Sensor PCB Design Guidelines*.

7 PACKAGE DIMENSIONS

Package dimensions for the ICP-10125:



Top View



Side & Bottom View

Figure 6. ICP-10125 Package Diagram

Recommended PCB land pattern for the ICP-10125:

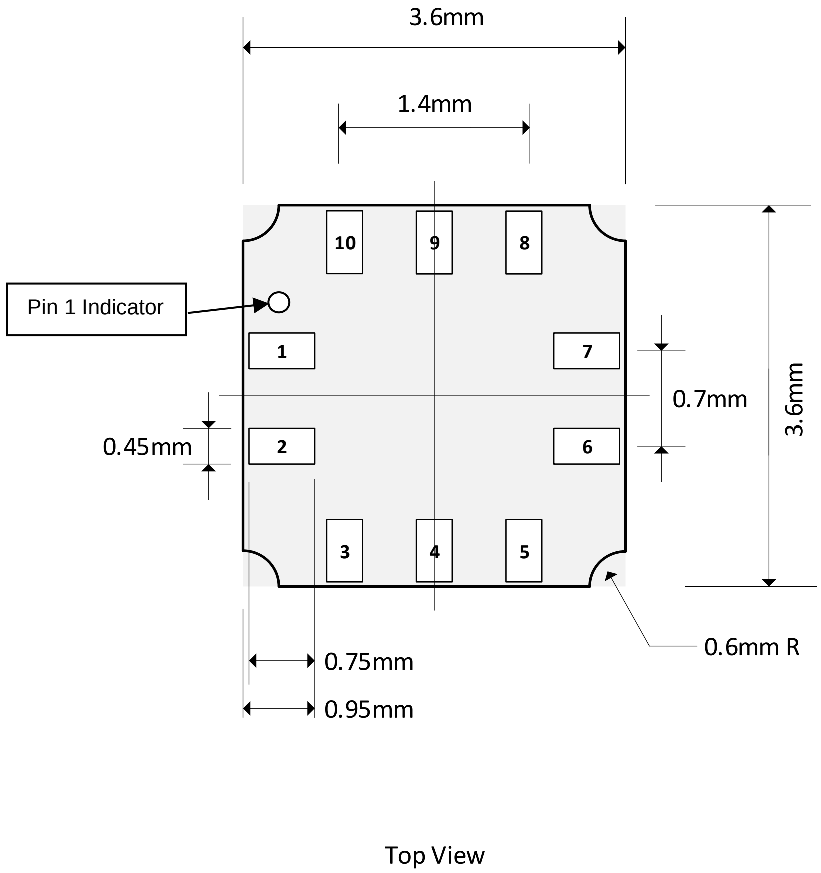


Figure 7. ICP-10125 recommended PCB land pattern

Product artwork for the ICP-10125:

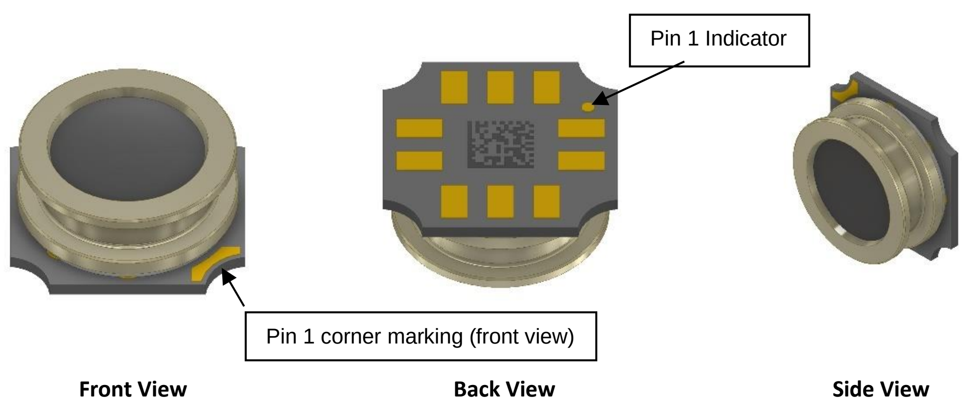


Figure 8. ICP-10125 Artwork

8 TAPE AND REEL SPECIFICATION

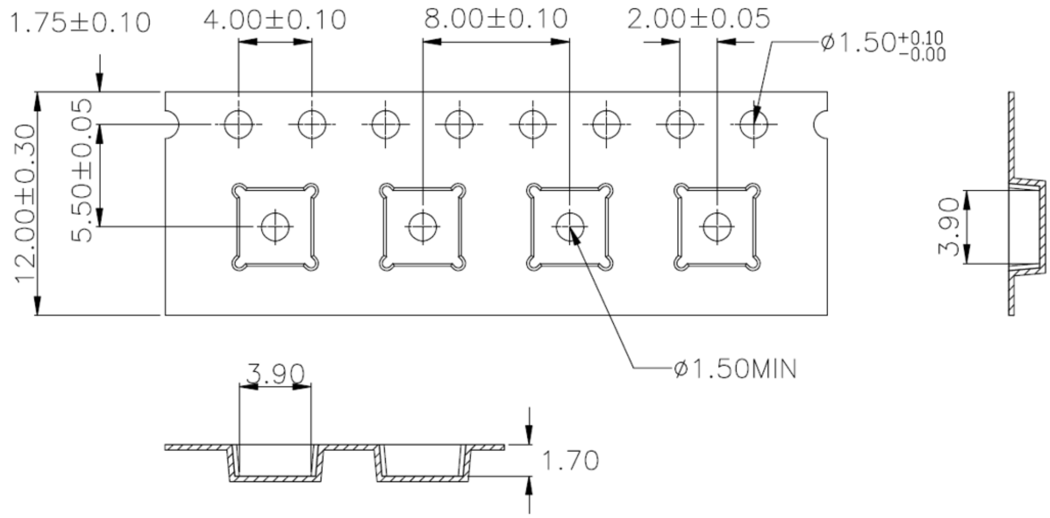


Figure 9. ICP-10125 Tape Dimensions

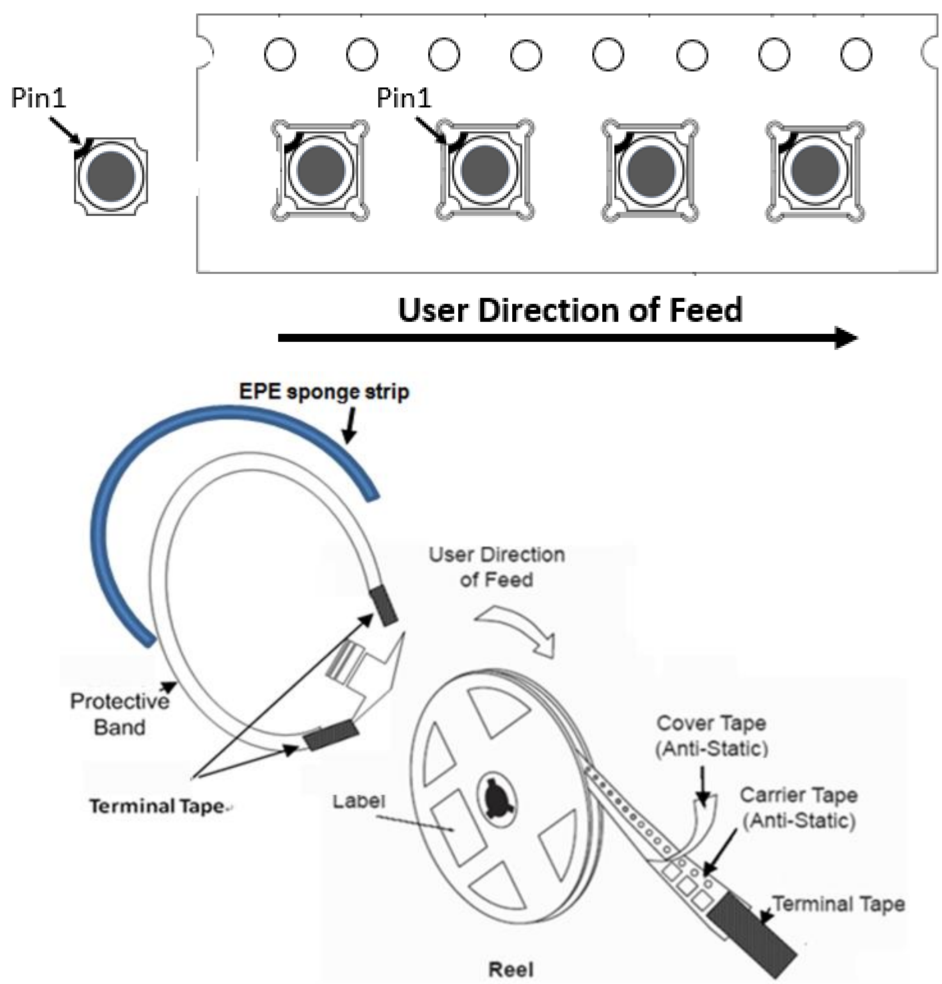


Figure 10. Tape and Reel Orientation

9 ORDERING GUIDE

PART	TEMP RANGE	PACKAGE BODY	QUANTITY	PACKAGING
ICP-10125†	−40°C to +85°C	3.55x3.55x1.45mm HTCC-10L	3000	13” Tape and Reel

†Denotes RoHS and Green-Compliant Package

10 REFERENCES

Please refer to “InvenSense MEMS Handling Application Note (AN-IVS-0002A-00)” for the following information:

- Manufacturing Recommendations
  - Assembly Guidelines and Recommendations
  - PCB Design Guidelines and Recommendations
  - MEMS Handling Instructions
  - ESD Considerations
  - Reflow Specification
  - Storage Specifications
  - Package Marking Specification
  - Tape & Reel Specification
  - Reel & Pizza Box Label
  - Packaging
  - Representative Shipping Carton Label
- Compliance
  - Environmental Compliance
  - DRC Compliance
  - Compliance Declaration Disclaimer

11 REVISION HISTORY

REVISION DATE	REVISION	DESCRIPTION
09/04/2020	1.0	Initial Release
04/09/2021	1.1	Formatting Updates; Updated Pressure Sensor Specs (Table 3); Added Tape and Reel Specification (Section 8)